Local Clock Glitching Fault Injection with Application to the ASCON Cipher

Sandeep G. Surya, Paolo Maistri, S. Sankaran

Local Clock Glitching Fault Injection with Application to the ASCON Cipher

G Surya[1], Paolo Maistri[2], and Sriram Sankaran[1]

[1] Center for Cyber Security Systems and Networks, Amrita Vishwa Vidyapeetham, Amritapuri, India
suryag@am.students.amrita.edu, srirams@am.amrita.edu
[2] Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, 38000 Grenoble, France
paolo.maistri@univ-grenoble-alpes.fr

Abstract. Lightweight ciphers such as ASCON facilitate ease of implementation as well as provide better performance over conventional ciphers, thus making it suitable for resource-constrained devices. However, hardware implementations of these ciphers are vulnerable to a multitude of physical attacks (such as fault injections) requiring dedicated countermeasures thus causing a negative impact on security, performance and power consumption. Modeling and understanding the impact of these attacks on cipher operations and end users is mandatory. Further, detection and mitigation of such fault injection attacks is challenging due to the interconnected nature of cipher design, coupled with the varying number of possible design choices and with the forced trade-offs that need to be done in order to implement expensive countermeasures at the lowest possible cost. In this work, we aim to model a fault injection attack on ASCON and analyze its impact on FPGA. In particular, we implement the ASCON cipher and propose a methodology for fault injection attacks using synchronous clock glitching by Digital Clock Manager (DCM) introducing a novel approach of locality, which can be exploited to emulate general delay faults on focused parts of the design, such those induced by pulsed EM injections.

Keywords: Lightweight cryptography · ASCON · fault injection attack · clock glitching.

1 Introduction

Lightweight ciphers address the need of resource-constrained devices by providing better performance as well as facilitating ease of implementation. In particular, they can be implemented to run faster on hardware or software and scale well resulting in higher throughput in limited-resource contexts, where conventional standards (e.g., AES) would struggle to guarantee acceptable performance or would require unfeasible costs [1]. Typically, lightweight ciphers utilize classical cryptographic algorithms that are modified and adapted to the hardware features and limitations at a low cost [2]. More recently, contests promoted by international standard organisations (such as the CAESAR contest) aim at establishing

a common ground to evaluate different proposals and define a shared standard for lightweight cryptography. Among the solutions, the ASCON cipher [3] is one of the most popular due to its versatility, efficiency, and performance. ASCON is a block cipher family of authenticated encryption and hashing algorithms designed for being lightweight and simple to implement, even with additional countermeasures against side-channel attacks. In March 2018, the ASCON authenticated cipher was chosen as a finalist and was confirmed in February 2019 by the CAESAR committee as the first choice for the case of lightweight use [4]. ASCON asserts a balanced design ideal for lightweight systems on both hardware and software, and can be efficiently protected against side channel analysis (SCA) through effective application of proper countermeasures.

ASCON cipher shows promise towards low-power devices, which are nonetheless vulnerable to a multitude of threats at different levels of the device, where attacking the cryptographic implementation can give access to sensitive information within the system. For instance, attackers can mount a clock glitching attack which causes unintended behavior of the ASCON cipher. Further, the design of the ciphers is composed of different stages that are typically interconnected such that injection of fault in a particular stage can cause a chain effect in the remaining stages thus corrupting the output bits. More advanced fault attacks are based on laser illumination, which allow the attacker to precisely tune location and timing, but are on the other hand rather expensive and require complex preparation of the targeted device. More recently, Electromagnetic (EM) injections have gained popularity, as they are simpler to mount while providing good timing and spatial tuning.

There exists a need for modeling and understanding the impact of attacks and related countermeasures on cipher operations and the consequences for the end users. This is of paramount importance due to the increasing sensitivity of the data coupled with advanced capabilities of attackers. For instance, a fault injection attack can range from a simple clock [5], or power [6] glitching, to a much more advanced execution of error injections of multiple order: multiple synchronized error injections on different parts of the circuit, or repeated injections over time. Similarly, side-channel attacks are composed of simple and advanced methods aimed at extracting the sensitive information processed in the circuit, from the simple analysis of power and EM traces [7], to differential analysis of multiple order, to mutual information analysis [8] or template attacks [9]. Additionally, the exploitation of both techniques at the same time has also been proposed, in order to extract information from side channels when faults are occurring. Each of these attacks produce a varying impact that needs to be measured, in order to assess correctly the threats and the implications of the attack vectors in terms of the performance and power consumption.

In this work, we implement the ASCON lightweight cipher using VHDL language on a FPGA device, and propose a fault injection methodology with application to the ASCON cipher, but the approach can be easily generalized to other cryptographic architectures. While based on well-known clock glitching, our methodology emphasizes spatial locality rather than global faulting, thus

limiting the propagation of errors to small and controlled parts of the design. As a perspective, this methodology can be also leveraged to model generic local delay faults, such as those induced by electromagnetic pulsed injections. It is known in fact, that pulsed EM injections can lower the power supply margin by coupling with ground and power rails, thus locally slowing the switching delay of the targeted gates [1]. Finally, we analyze the power, energy, throughput and area of the ASCON-128 and examine the impact of fault injection attacks on the ASCON implementation.

In the next section, we recall the theoretical background on ASCON and related works on hardware implementation of the cipher as well as on fault injections. We present our methodology in Section 4 and the experimental setup, together with the preliminary results, in Section 5. Finally, we discuss some possible countermeasures in Section 6.

2 Background

2.1 Description of ASCON

ASCON is a symmetric key algorithm for authenticated encryption and the mode of operation is based on sponge duplex modes. The ASCON encryption is updated in four steps: Initialization, Associated data Processing, Plaintext / Ciphertext Processing, and Finalization.The same permutation function is used in all the steps. ASCON has a strong permutation function that takes place between the initialization and the finalization stage. The permutation function consists of a round-dependent constant addition, a substitution layer, and a diffusion layer. The cipher has five inputs which consist of: a variable-length plain text, a variable-size associated data, a fixed-length secret message (which can also be zero), a constant-size public message, and a fixed-size key.

ASCON uses a cryptographic duplex construction for encryption and a sponge construction for the absorption of the associated data. The associated data and the plaintext are XOR'ed with a portion of the permutation function, which is then inserted into the next iteration of rounds. This offers secrecy and integrity at once. The internal state contains 5 substates of 64-bit, giving 320-bit total word length. The associated data and plaintext are processed after the initialization in 128-bit registers. The Finalization begins after generation of the ciphertext. The final phase output is the 128-bit tag T.

The decryption method is carried out according to the reversed encryption method, while the latter transformations have separate concepts that support the encryption transformations. The key difference is that the ciphertext block is explicitly placed into the permutation function, and the plaintext is taken from the XOR output of proceesing ciphertext phase. Thus, both encryption and decryption can be easily implemented at once with minimal effort. As hinted briefly above, the permutation feature of ASCON involves a constant insertion, the S-box (the substitution) and a diffusion layer. The S-box is a 5-bit, non-linear function. The S-box inputs are five state terms, one bit per word. There

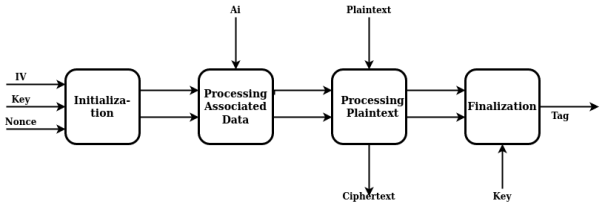

Fig. 1. ASCON encryption

are, in principle, 64 S-boxes that fit 64 bits of the word state. The modified data at the S-box feature output is then passed via the linear diffusion layer.

2.2 Our Implementation

Our ASCON-128 design (see Figure 2) has been implemented using VHDL on FPGA device. The practical verification simulations of these designs have been carried out with Mentor Graphics ModelSim. The initial input to the ASCON cipher is five 64 bit states and the state is composed of five gated clock shift registers with independent shift enabled inputs. All state registers are activated and shifted bit-slice-wise through the single S-box instance for calculating the S-box. Plaintext is loaded in the first clock cycle, and then all intermediate states are computed in the next cycles; data is processed through multiplexer in order to correctly route the information at the right time. First, data is stored in the Encryption Register and combined with the round constant. Next, the mixed state is transferred to the S-box layer, and then the linear diffusion layer (which supplies 64-bit data concurrently to the p-layer) and is finally transmitted through the multiplexer back to the Encryption Register. In the last clock cycle, the cipher text and the tag is obtained from the finalization phase.

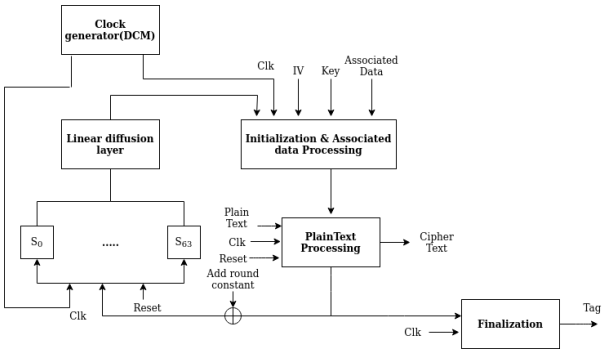

Fig. 2. Proposed diagram of Hardware Implementation of ASCON

3 Related Work

Gross et al. [4] proposed one of the first hardware implementations for the CAESAR candidate ASCON. In particular, they present ASCON hardware implementations for RFID tags, Wireless Sensor Nodes, Embedded Systems, and applications that need optimum performance. In this work they prove that ASCON is not only fast and small, but can also be easily protected against first-order DPA Attacks. They also propose a threshold-based implementation of ASCON to protect from side-channel attacks.

Ghalaty et al. [1] introduce the implementation of Differential Fault Intensity Analysis techniques on PRESENT and LED implementations. It is based on the assumption that the errors are biased, and thus not evenly distributed around the cipher state variables. Keyvan Ramezanpour et al. [9] introduce the fault injection approach against ASCON. In this paper, they examine the vulnerability of authenticated ASCON against fault attacks. Most Fault Attacks (FA) techniques use the information leaked as a result of errors in intermediate variables caused by injections of faults. However, the introduced statistical ineffective fault analysis (SIFA), based on a reduced fault model, utilizes the probability that a fault injection does not cause errors – the so-called unsuccessful fault induction. Fan Zhang et al. [14] propose a feature of persistence applied to fault attacks termed the persistent fault attacks. The Persistent Fault Analysis (PFA) is proved on various implementation of AES-128, particularly fault-hardened implementation based on Dual Modular Redundancy (DMR). The solution proposed shows the feasibility and practicability of the attack, a case study with a row hammer technique is demonstrated on the shared library Libgcrypt.

Threshold implementations one of the defending method, as they have been recently proposed to address both side channel and fault attacks. Thomas De Cnudde et al. [15] present a threshold implementation of the Advanced Encryption Standard S-box which is safe against power analysis attacks of the first and second order. They propose a higher order threshold implementation of AES to defend against all the fault injection attacks. Hannes Gross et al. [16] described the efficient method to defend against side-channel attacks. They demonstrate how the masking algorithm called Unified Masking (UMA) can be used to protect hardware implementations. Schneider, Moradi et al. [17] introduced in 2016 a countermeasure for cryptographic hardware implementations that combines the concept of a provably-secure masking scheme i.e., threshold implementations, with an error detecting approach against fault injection.

The purpose of our research is to investigate the effect of clock glitches on ASCON lightweight cipher. For this we implemented ASCON on an FPGA device and supply the synchronous clock using Digital Clock Manager (DCM) modules. In order to inject an error in the ASCON computation, we generate a clock glitch in an S-box of ASCON. Our research goal is to analyse the effects of fault injections, taking our hardware implementation of the lightweight cipher as use case.

4 Proposed Methodology

The hardware used here as fault injection evaluation board is the SASEBO-GII, shown in Figure 3: the reason for choosing this FPGA board as a hardware implementation platform is that this board was specifically designed for security assessment and fault injection experiments of cryptographic implementations. Unlike non-reconfigurable devices such as ASIC, the FPGA device have high potential and versatility. FPGA is a logical hardware device that combines of several logic cells and programmable switches. The logic cells include slice registers, look up tables(LUT), flip-flops and the programmable switches designed to communicate between the logic elements. In this paper, we have proposed an efficient ASCON architecture to increase performance and make less use of hardware so we implemented the pipeline technique where the ASCON cipher is divided into multile phases and each phase results is stored into corresponding registers.

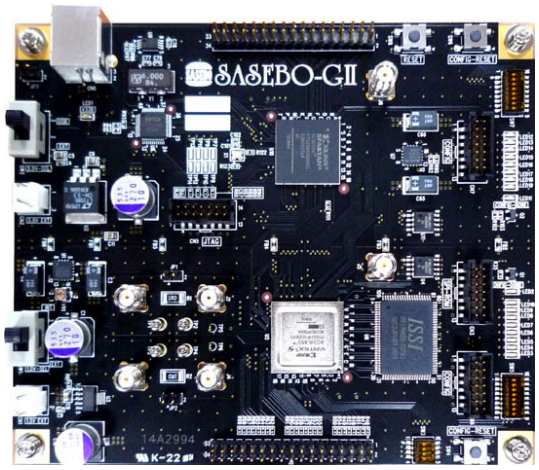

Fig. 3. SASEBO-GII board

In the proposed design we use a synchronous clocking circuit,in this approach all the flip-flops use the same clock input and registers may latch the data at the rising edge of the clock [18]. The Digital Clock Manager (DCM) module is used to produce synchronous clock in FPGA [19]. We developed a dedicated method for implementing clock signal glitches. In our proposed hardware implementation, the Clk fed to ASCON drives the output of the FPGA by integrating by reference signal Clk(nominal). Glitches are injected into the S-box through high frequency faulty signal(Clk2x) shown in figure 4. Both nominal and high-frequency clock signals are aligned in their phase for precision and reproducibility of the experiments.

Here, we show how to build a precise glitch at right and controllable position attack on the ASCON. To create a glitched clock, we use Xilinx FPGA's global clock buffers provided by DCMs, as shown in Figure 4. The Global Clock Buffers have three distinct characteristics. These may also be programmed as a regional clock buffer with a clock enabler (BUFGCE) as well as the single clock delivery. The clock can then be stopped at the clock buffer output at any point. A synchronous 2:1 multiplexer (BUFGMUX) clock buffer can be configured. Such multiplexers can switch between two clock sources at any time using the select input that user logic can control. There is no need for unique phase relations between the two clocks. A BUFGMUX powered by a couple of DCMs is connected to the clock network. DCMs monitor clock signal delays with the phase-shift parameter. The Clk signal indicates the on-board clock signal, Clk2x is the high frequency signal, Glitched Clock is the faulty glitched signal supplied by the DCM. The phase shift parameters to DCM monitor the time periods of the proposed design. We evaluate the basic properties of the proposed glitch generator implemented in FPGA. A glitchy global clock cycle can be induced into the S-box of ASCON.

The delay line includes a delayed version of the input clock Clk. The DLL in the DCM adds a delay between the input clock and the feedback clock until both clocks are in the same phase. There are two S-box designs: one with the usual clock and therefore the correct output, and the other with the glitchy clock and faulty output. In the S-box block of ASCON, we inject the clock glitch. Then we compare the correct execution and faulty executions critical path delay and power and area.

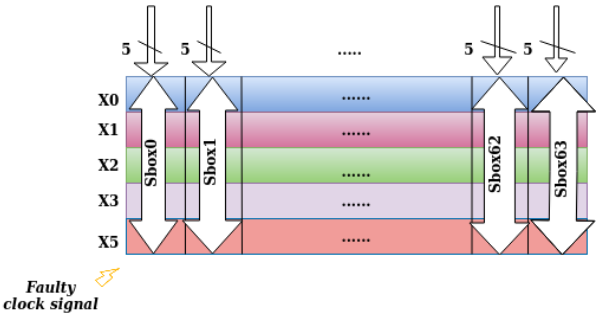

Fig. 4. Clock glitching applied on the S-box of ASCON

In our proposed attacking methodology, the glitch is injected in the S-box of ASCON in every encryption of algorithm. ASCON's initial state is 320 bit data that divided into five 64 bit data x0, x1, x2, x3, x4 these data is given as input to S-box. After the S- box operation it subjected to nonlinear process uses 5-bit S-boxes that change the state with a single bit of every state word with the data input and perform shifting operation in each output of the S-box. S-box

output is accompanied by non-uniform distribution of the data. The glitched clock makes faults in the S-box operations resulted to faulty output in the linear diffusion operations.

It is important to emphasize that our approach is not targeted at feeding a global glitched clock at the whole design as it is generally done in the state of the art, but rather partition the design and apply the faulty clock just to a smaller part of the circuit. The motivation is twofold: first, we can analyze more finely the error distribution and propagation with respect to each targeted functional block of the design, and second, controlling and limiting the locality allows emulating the effects of EM injections without recurring to the actual experiments, that become thus accessible even without the proper equipment.

5 Evaluation

The device utilization summary of hardware implementation of ASCON and fault injected ASCON is given in Table 1. From the table, fault injected ASCON consumes significant resources than ASCON. Power consumption of fault injected ASCON was observed to be nearly equal to ASCON. In our design, clocks are generated by global clock network driven by global clock multiplexers (BUFGMUX) that can multiplex between two global clock sources or be used as a simple BUFG clock buffer. The first encrypted data block of ASCON cipher takes latency of 85 clock cycles.

Table 1. Design Utilization Summary of ASCON and fault injected ASCON

Elements	Available Resources	ASCON	Fault-injected ASCON
No. of Slice registers	44800	206	210
Slice LUTs	44800	211	217
Bonded IOBs	640	201	201
Power(mW)	-	7.7	7.8
Throughput (Mbps)	-	211	198

Figure 5 shows the delay comparison between ASCON and its fault injected version in each of the rounds. For the basic ASCON implementation, the latency of each round is between 0.3s-0.5s; on the other hand, the implementation augmented with clock glitching injection attacks shows delay nearly equals to gate delay of basic ASCON, as its latency ranges from 0.4-0.55. This effect is occurring because of the additional signal transitions taking place due to the clock glitching. If a glitched clock signal passes through gates, flip-flops and Look Up Tables (LUTs), clock skew takes place in the circuit that leads to the delay in the circuit controlling the clock glitch through DCM. In a design driven by a single clock at the lowest common frequency, the actual overhead is limited to about 10%, which can be considered as acceptable.

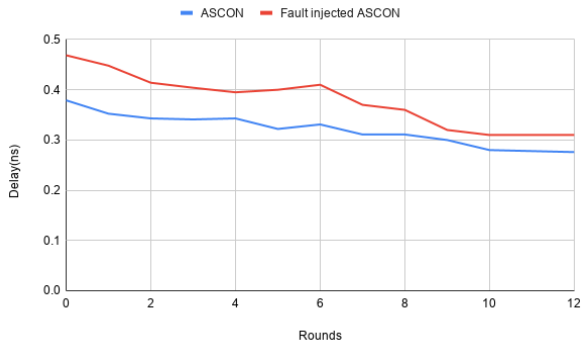

Fig. 5. Comparison of delay in ASCON and fault injected ASCON

Figure 6 shows the fault distribution in each round of ASCON-128, obtained by collecting and classifying the fault distribution in each of the rounds. We observed that the fault distribution varies in each of the rounds. The glitch cannot be recognized for glitch periods of less than a certain amount (4.1 ns) because the shape of the glitch signal is masked due to LUT filtering. Moreover, the length interval of [4.1 ns, 5.4 ns] produces incorrect output bytes and values over 5.4 ns do not affect registers value in the prposed design.

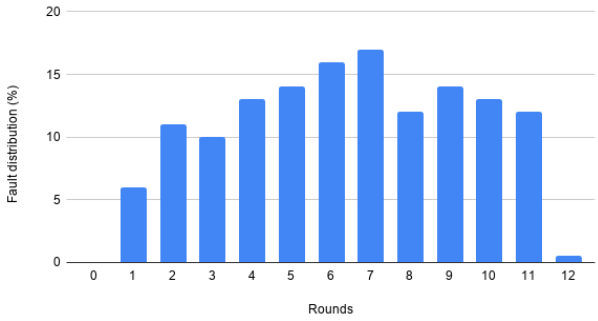

Fig. 6. Fault distribution of each rounds of ASCON

6 Perspectives on Possible Countermeasures

6.1 Threshold Implementation

A threshold Implementation is a strategy to secure hardware implementations from first-order attacks [21] [22]. The number of necessary shares, N is the algebraic degree of non-linear function plus one to defend against first-order DPA.

There are three criteria for a threshold implementation (TI): non-completeness, reliability, and accuracy. On each share, linear steps can be determined separately and eventually recombined. Shares must be mixed in the non-linear steps. If at most N-1 shares are combined at once, an implementation is not complete.

The basic theory behind TI is that the measurements do not take effect explicitly on sensitive security details, but indirectly by modified transformations on so-called equities. Linear transformations may be carried out independently for any component such as the inclusion of a round constant or the linear layer of ASCON. The S-box variable functions need to satisfy the properties of consistency, non-completeness, and uniformity in order to withstand first order DPA attacks [22]. Threshold design has shown broad applicability in a large variety of crypto algorithms, ranging from symmetric algorithms to asymmetric algorithms that have been effectively protected.

6.2 Unified Masking Approach

Unified masking Approach (UMA) is also an effective method to defend against fault injection and side channel attacks [16]. In ASCON, the core consists of the Finite State Machine (FSM) and the round counter driving the required control signals and the state unit that forms the datapath of knowledge where all state transformations are computed. The state module has a separate dedicated FSM and carries out four sub-steps of round transformation. The registers are in the S-box configuration in the ASCON system. Pipeline registers are shared between the AND gates. For glitch resistance, registers in State 0 after the XOR gate are critical. Due to the fact that the masking AND Gate combines behavior from numerous fields, d-probing resistance will fail. Notice that XOR output gates should not be recorded in addition since they are included in one of the state registries. Therefore monitoring security is given as long as no common areas have been crossed during the linear portions of this process. We ensure that each component and section of the circuit is aligned with a specific mutual domain which is maintained in the circuit. In ASCON algorithm, the most vulnerable part is the substitution layer (S-box): in Unified Masking Approach, pipeline registers are placed to avoid the fault injection in the S-box design and make the design more secure. Unified Masking Approach(UMA) appears to be the most appropriate approach for hardware applications.

6.3 Comparison of Threshold Implementation and Unified Masking Approach

Unified Masking Approach is an efficient framework to protect against fault injection attacks and offers a higher degree of security for the cryptographic algorithms. On the other hand, Threshold based implementation is also used to secure the hardware designs. Yet, a high degree of resistance correlates with the increasing need for fresh randomness, which significantly raises implementation costs. The costs of installation of each masking scheme thus rely heavily on

two factors: first, the amount of shares (or masks) available to cover the dth-order, and secondly, the cost of modification for nonlinear feature evaluation. By comparing both the approaches, high degree of security can be provided by Unified masking approach due to the reduction in the asymptotic randomness costs. Thus it becomes necessary to comparatively evaluate the resilience of both of the approaches against fault injection attacks coupled with assessing their corresponding overheads.

7 Conclusion

In this work, we developed a clock glitch fault injection attack on ASCON algorithm implemented on FPGA. To achieve this goal, we implemented ASCON and analysed the power, throughput, and area of the proposed design. Further, we modeled a fault injection attack using clock glitching and showed that faults can be injected at certain frequencies; we analysed the power, area, throughput overheads with respect to the basic implementation, and the varying fault distribution effects for each round. The proposed attack methodology successfully injects faults in ASCON as validated by our experiments on FPGA. In the future, we plan to exploit this methodology to emulate EM fault injection, as well as address the discovered vulnerabilities by implementing a subset of suitable countermeasures.

References

1. Ghalaty, N.F., Yuce, B. and Schaumont, P., 2015, April. Differential fault intensity analysis on PRESENT and LED block ciphers. In International Workshop on Constructive Side-Channel Analysis and Secure Design (pp. 174-188). Springer, Cham.
2. Park, JeaHoon, SangJae Moon, DooHo Choi, YouSung Kang, and JaeCheol Ha. "Differential Fault Analysis for Round-Reduced AES by Fault Injection." ETRI Journal 33, no. 3 (2011): 434-442.
3. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, Martin Schläffer. Ascon v1.2. Submission to Round 3 of the CAESAR competition (2016).
4. Groß, Hannes, Erich Wenger, Christoph Dobraunig, and Christoph Ehrenhöfer. "Suit up!–Made-to-Measure Hardware Implementations of ASCON." In 2015 Euromicro Conference on Digital System Design, pp. 645-652. IEEE, 2015.
5. Roscian, C., Dutertre, J.M. and Tria, A., 2013, June. Frontside laser fault injection on cryptosystems-Application to the AES'last round. In 2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST) (pp. 119-124). IEEE.
6. Patranabis, S., Chakraborty, A., Nguyen, P.H. and Mukhopadhyay, D., 2015, April. A biased fault attack on the time redundancy countermeasure for AES. In International workshop on constructive side-channel analysis and secure design (pp. 189-203). Springer, Cham.
7. Korczyc, J. and Krasniewski, A., 2012, April. Evaluation of susceptibility of FPGA-based circuits to fault injection attacks based on clock glitching. In 2012 IEEE 15th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS) (pp. 171-174). IEEE.

8. Skorobogatov, S., 2011. Fault attacks on secure chips. Design and Security of Cryptographic Algorithms and Devices.
9. Ramezanpour, Keyvan, Paul Ampadu, and William Diehl. "A statistical fault analysis methodology for the ascon authenticated cipher." In 2019 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), pp. 41-50. IEEE, 2019.
10. Dehbaoui, Amine, Jean-Max Dutertre, Bruno Robisson, and Assia Tria. "Electromagnetic transient faults injection on a hardware and a software implementations of AES." In 2012 Workshop on Fault Diagnosis and Tolerance in Cryptography, pp. 7-15. IEEE, 2012.
11. Clavier, C., Feix, B., Gagnerot, G., Roussellet, M. and Verneuil, V., 2011, September. Improved collision-correlation power analysis on first order protected AES. In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 49-62). Springer, Berlin, Heidelberg.
12. Archambeau, C., Peeters, E., Standaert, F.X. and Quisquater, J.J., 2006, October. Template attacks in principal subspaces. In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 1-14). Springer, Berlin, Heidelberg.
13. Ramezanpour, Keyvan, Paul Ampadu, and William Diehl. "FIMA: Fault Intensity Map Analysis." In International Workshop on Constructive Side-Channel Analysis and Secure Design, pp. 63-79. Springer, Cham, 2019.
14. Zhang, Fan, Xiaoxuan Lou, Xinjie Zhao, Shivam Bhasin, Wei He, Ruyi Ding, Samiya Qureshi, and Kui Ren. "Persistent fault analysis on block ciphers." IACR Transactions on Cryptographic Hardware and Embedded Systems (2018): 150-172.
15. De Cnudde, T., Bilgin, B., Reparaz, O., Nikov, V. and Nikova, S., 2015, November. Higher-order threshold implementation of the AES S-box. In International Conference on Smart Card Research and Advanced Applications (pp. 259-272). Springer, Cham.
16. Gross, Hannes, and Stefan Mangard. "A unified masking approach." Journal of cryptographic engineering 8, no. 2 (2018): 109-124.
17. Moradi, A., Poschmann, A., Ling, S., Paar, C. and Wang, H., 2011, May. Pushing the limits: A very compact and a threshold implementation of AES. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 69-88). Springer, Berlin, Heidelberg.
18. Deepakumara, J., Heys, H.M. and Venkatesan, R., 2001, May. FPGA implementation of MD5 hash algorithm. In Canadian Conference on Electrical and Computer Engineering 2001. Conference Proceedings (Cat. No. 01TH8555) (Vol. 2, pp. 919-924). IEEE.
19. Brynjolfson, I. and Zilic, Z., 2000. FPGA clock management for low power. In Proceedings of the International Symposium on FPGAs (pp. 219-219).
20. Moradi, A., Poschmann, A., Ling, S., Paar, C. and Wang, H., 2011, May. Pushing the limits: A very compact and a threshold implementation of AES. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 69-88). Springer, Berlin, Heidelberg.
21. Bilgin, B., Gierlichs, B., Nikova, S., Nikov, V. and Rijmen, V., 2014, May. A more efficient AES threshold implementation. In International Conference on Cryptology in Africa (pp. 267-284). Springer, Cham.
22. Sugawara, T., 2019. 3-share threshold implementation of AES S-box without fresh randomness. IACR Transactions on Cryptographic Hardware and Embedded Systems, pp.123-145.